(19) JAPANESE PATENT OFFICE

PATENT ABSTRACTS OF JAPAN

(11) Publication number: **08305563 A**

(43) Date of publication of application: **22 . 11 . 96**

(51) Int. Cl **G06F 9/32**

(21) Application number: **07112767**

(22) Date of filing: **11 . 05 . 95**

(71) Applicant: **HITACHI LTD**

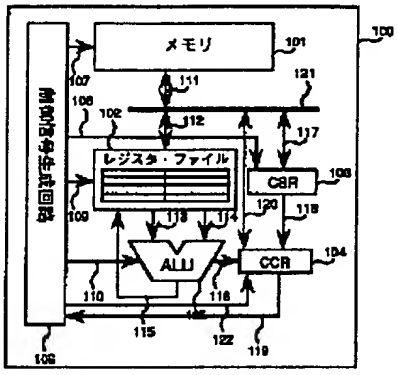
(72) Inventor: **KIUCHI ATSUSHI**
HATANO YUJI

(54) DATA PROCESSING UNIT

(57) Abstract:

PURPOSE: To provide the data processing unit to support a conditional arithmetic instruction in which the processing efficiency is improved without causing problems such as performance deterioration due to a limit of a degree of freedom of operands or kinds of instructions and increase in program memory capacity due to increase in an instruction code length.

CONSTITUTION: The processing unit is provided with a condition selection means selecting at least one condition among plural conditions to be used for a conditional arithmetic instruction, a conditional information storage means 103 (CSR) storing condition information selected by the condition selection means, and a means 104 (CCR) storing information denoting property of the result of arithmetic operation based on at least the selected condition and the conditional arithmetic instruction has a function to execute required arithmetic processing one of selected conditions is designated and the condition is established.



(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号

特開平8－305563

(43)公開日　平成８年(1996)11月22日

(51)Int.Cl.[6]	識別記号	庁内整理番号	ＦＩ	技術表示箇所
Ｇ０６Ｆ　9/32	３２０		Ｇ０６Ｆ　9/32	３２０Ｆ

審査請求　未請求　請求項の数６　ＯＬ　（全 11 頁）

(21)出願番号　特願平7－112767

(22)出願日　平成７年(1995)５月11日

(71)出願人　000005108
株式会社日立製作所
東京都千代田区神田駿河台四丁目６番地

(72)発明者　木内　淳
東京都小平市上水本町五丁目20番１号　株式会社日立製作所半導体事業部内

(72)発明者　波多野　雄治
東京都小平市上水本町五丁目20番１号　株式会社日立製作所半導体事業部内

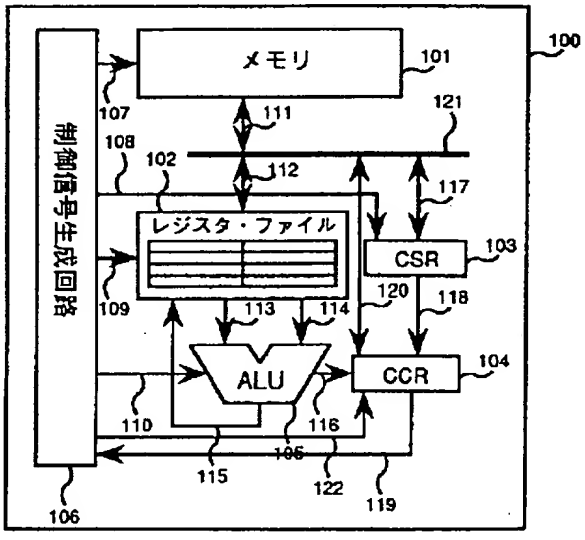
(74)代理人　弁理士　磯村　雅俊

(54)【発明の名称】　データ処理装置

(57)【要約】

【目的】　命令の種類やオペランドの自由度の制限による性能低下または命令コード長の増大によるプログラムメモリ容量増加などの問題を発生することなく処理効率の向上を図ることができる条件付き演算命令をサポートするデータ処理装置を提供すること。

【構成】　条件付き演算命令に使われるべき複数の条件から、少なくとも１つの条件を選択する条件選択手段と、該条件選択手段で選択された条件情報を保持する条件情報保持手段１０３（ＣＳＲ）と、少なくとも前記選択された条件に基づいて演算結果の性質を表す情報を保持する手段１０４（ＣＣＲ）を具備し、該条件付き演算命令は該選択された条件のうちのひとつを指定して条件が成立しているときのみ、所要の演算処理を実行する機能を有する。



1

【特許請求の範囲】

【請求項１】　データ処理を実行する演算回路と、該データ処理に必要なソースデータを供給する手段と、演算結果を格納する演算結果格納手段と、指定された条件が成立している時のみ演算を実行する条件付き演算命令を実行する制御手段とを少なくとも具備するデータ処理装置において、該条件付き演算命令に使われるべき複数の条件から少なくとも１つの条件を選択する条件選択手段と、該条件選択手段で選択された条件情報を保持する条件情報保持手段と、少なくとも該選択された条件に基づいて演算結果の性質を表す情報を保持する手段をさらに具備し、該条件付き演算命令は該選択された条件のうちのひとつを指定して条件が成立しているときのみ、所要の演算処理を実行する機能を有することを特徴とするデータ処理装置。

【請求項２】　前記演算結果の性質を表す情報を保持する保持手段は、前記選択された条件に基づいて演算結果の性質を表す情報以外に恒常的に演算結果の性質を表す情報を同時に保持する手段を持つことを特徴とする請求項１記載のデータ処理装置。

【請求項３】　前記条件選択手段は、恒常的に演算結果の性質を表す複数の情報の中からひとつを選択することを特徴とする請求項１記載のデータ処理装置。

【請求項４】　前記条件付き演算命令は、命令コード中に条件を指定するフィールドを持ち、該条件を指定するフィールドは、無条件で実行する選択肢と、前記選択された条件情報を保持する手段で保持されている条件が真の場合のみ所要の演算処理を実行する選択肢を少なくとも持つことを特徴とする請求項１記載のデータ処理装置。

【請求項５】　前記条件付き演算命令は、命令コード中に条件を指定するフィールドを持ち、該条件を指定するフィールドは、無条件で実行する選択肢と、前記選択された条件情報を保持する手段で保持されている条件が「真」の場合に所要の演算処理を実行する選択肢と、「偽」の場合に所要の演算処理を実行する選択肢とを少なくとも持つことを特徴とする請求項１記載のデータ処理装置。

【請求項６】　前記選択された条件情報を保持する手段は、前記演算結果の性質を表す情報を保持する手段と同一のレジスタで構成されていることを特徴とする請求項１記載のデータ処理装置。

【発明の詳細な説明】

【０００１】

【産業上の利用分野】本発明は、条件付き演算命令をサポートするデータ処理装置に関し、特に、条件付き演算命令の分岐条件指定フィールドに必要なビット数を低減することによって、命令の種類やオペランドの自由度の制限による性能低下および命令コード長の増大によるプログラムを格納するメモリ容量の増加を抑えることが可

2

能なデータ処理装置に関する。

【０００２】

【従来の技術】近年の高度に発展した情報処理社会においては様々な分野でディジタル信号処理が行われており、そのためのハードウエアとして、通常マイクロプロセッサやディジタルシグナルプロセッサ（ＤＳＰ）と呼ばれるシステムＬＳＩが使われている。これらのシステムＬＳＩは、様々な用途に適用させるために、汎用の命令セットを備えているのが普通である。様々な用途のうちには、一連の固定した命令の順序に従って実行するだけでよい単純な処理もあるが、成立する条件によって実行すべき処理内容が異なってくるのが一般的である。その場合には、まず所定の条件が成立するか否かを判定し、その判定結果によって処理すべき命令を選択する必要がある。そのため従来の一般的なシステムＬＳＩは条件分岐命令を備えている。図１５に条件分岐命令による分岐処理のフローチャート例を示す。

【０００３】図１５では、まず、ステップ１５０で所定の算術演算１を実行し、次にステップ１５１で前ステップ１５０による演算結果が正の数（０を含む）か否かを判定する。その判定結果、負の数であった場合（NO）には直接ステップ１５３に移り動作３を実行するが、ステップ１５１による判定結果が正の数（０を含む）の場合（YES）にはステップ１５２で動作２を実行した後、ステップ１５３に移る。すなわち、図１５の処理例ではステップ１５０での実行結果によって以降の処理手順が異なっている。この処理をアセンブラ的なニモニックで表現したプログラム例を図１６に示す。

【０００４】通常のシステムＬＳＩでは、図１５のような条件によって処理手順が異なる処理を実現するために、条件分岐命令を用いる。すなわち、図１６に示すように１行目の算術演算命令inst.1を実行することにより、その演算結果を求める（図１５のステップ１５０に相当）。この時、通常のシステムＬＳＩでは同時にその演算結果の性質を表すいくつかのコードを生成する。これは通常にコンディション・コードと呼ばれており、例えば、桁上がりや桁借りが生じたことを示すフラグ（キャリー），結果が正か負かを示すフラグ（ネガティブ），結果がプロセッサが表現できるデータの範囲を超えたことを示すフラグ（オーバフロー），結果がゼロかどうかを示すフラグ（ゼロ）などがその代表なコンディション・コードのフラグ例である。

【０００５】図１６の２行目では、これらのコンディション・コードのうち、演算結果が正か負かを示すフラグを用いた条件分岐命令が書かれており、もし負であれば（ifN）ラベルL1のついている行に分岐する（図１５のステップ１５１の判定；NOに相当）。すなわち，もし負であれば３行目の命令は実行せずにスキップして４行目（図１５のステップ１５３に相当）に分岐する。もし条件判定の結果、正（０を含む）であれば、２行目の条件

分岐命令は実行されず、３行目のinst.3を実行（図１５のステップ１５２に相当）してから４行目に移る。このようにして図１５のフローチャートに示された処理が実行できる。

【０００６】ここで注目すべきことは、図１５のような処理を行う場合には必ず分岐動作が必要になることである。分岐命令の役割は基本的にはプログラムの実行順序を強制的に変更することであり、データ処理には直接には何ら寄与しない動作である。しかも処理性能を向上させる極めて一般的な手法であるパイプライン制御（先行制御）を行っているシステムＬＳＩでは、このような分岐命令によってプログラムの流れが乱されると、著しく処理効率が低下する（オーバヘッドが増加する）。このような処理効率の低下を極力抑えるために、従来より様々な手法が提案されている。一つの手段としては、この条件分岐命令の実行によるオーバヘッドを低減しようとするものがある。例えば条件が成立しても不成立でも同じように処理できるように、あらかじめ両方の場合で次に実行すべき命令をあらかじめプログラムメモリから読み出して用意しておく手法などが知られている。しかしこの方法はプログラムのシーケンス制御が非常に複雑になるため、ハードウェア量が増大してしまうという問題がある。

【０００７】このようなハードウェアの増大を抑えて処理性能を向上させる別の手段としてADSP-2100 User's Manual p4-4～4-15に開示されているように、様々な命令に分岐命令と同様な条件を付加し、指定された条件が成立した時のみ実行し、不成立の場合は指定された動作を実行せずにそのまま次のステップに移るようにしたものが知られている。この手法を用いて図１５の処理をアセンブラ的なニモニックで表現したプログラム例を図１７に示す。すなわち、図１５のステップ１５２はステップ１５０の結果が正の時のみ実行するのであるから、図１７の２行目（if P inst.2）に示されているように、１行目の演算結果として生成されるコンディション・コードのうちの正（０を含む）か負かを示すフラグが正（０を含む）を示している時のみ、inst.2を実行するような命令を備えることにより、分岐命令をプログラムから排除することができる。この２行目の命令は条件の成立／不成立に関わらず、プログラム実行のパイプラインは全く乱れないので、複雑な制御回路を増加することなしに処理性能の向上を図ることができる。

【０００８】

【発明が解決しようとする課題】しかしながら、上記の条件付き演算実行命令の手法では、複雑なパイプライン制御用のハードウェアは不要になるものの、様々な命令のコードに条件指定フィールドを設けなければならないため、限られた命令コードのうちの何ビットかがそのために使われてしまい、サポートする命令全体の種類やオペランドの自由度が狭くなり、プロセッサとしての性能が低下してしまうという問題がある。例えば、前述したような桁上がりや桁借りが生じたことを示すフラグ（キャリー），結果が正（０を含む）か負かを示すフラグ（ネガティブ），結果がプロセッサが表現できるデータの範囲を超えたことを示すフラグ（オーバフロー），結果がゼロかどうかを示すフラグ（ゼロ）の４種類のコンディションの真／偽を実行の条件として指定できるようにする場合、合計８種類の条件から選択することになるので、図１８に示すように命令コード９０毎に少なくとも３ビットの条件指定フィールド９１を確保する必要がある。３ビットの条件指定フィールド９１の具体例を９２に示してある。プロセッサとしての性能の低下を回避するために、この３ビットの条件指定フィールド９１を従来の命令コード長に新たに追加させると、今度はプログラム全体を格納するメモリ容量が増加してしまうという別の問題が生じてしまう。

【０００９】一般的にいって、上述したような条件付きで演算を実行すべきプログラムモジュールはそれほど頻繁には現れてくることはないが、例えば、ディジタル信号処理の場合には一定の処理ルーチンを繰り返し実行することが多く、処理ルーチン内に１行あるだけでもそれを何回も繰り返し実行するため条件付き演算命令の処理上の性能向上の効果が大きくなる。その上、所定の処理ルーチンを繰り返し実行する際には指定されている条件はずっと同一である。本発明は、この事情を鑑みてなされたものであり、その目的とするところは、上述した如き問題点を解決し、命令の種類やオペランドの自由度の制限による性能低下、および命令コード長の増大に伴うプログラムメモリ容量の増加などの問題を発生することなく処理効率の向上を図ることができる条件付き演算命令をサポートするデータ処理装置を提供することにある。

【００１０】

【課題を解決するための手段】本発明は、上記目的を達成するために、データ処理を実行する演算回路（図１の１０５（ＡＬＵ））と、該データ処理に必要なソースデータを供給する手段（同レジスタ・ファイル１０２）と、演算結果を格納する演算結果格納手段（レジスタ・ファイル１０２）と、指定された条件が成立している時のみ演算を実行する条件付き演算命令を実行する制御手段（制御信号生成回路１０６）とを少なくとも具備するデータ処理装置において、該条件付き演算命令に使われるべき複数の条件から、少なくとも１つの条件を選択する条件選択手段（例えば図４の１行目）と、該条件選択手段（例えば図４の１行目）で選択された条件情報を保持する条件情報保持手段（コンディション・セレクト・レジスタ１０３（ＣＳＲ））と、少なくとも前記選択された条件に基づいて演算結果の性質を表す情報を保持する手段（コンディション・コード・レジスタ１０４（ＣＣＲ））をさらに具備し、該条件付き演算命令は該選択

された条件のうちのひとつを指定して条件が成立しているときのみ、所要の演算処理を実行する機能を有することを特徴としている。

【００１１】また、前記演算結果の性質を表す情報を保持する保持手段（コンディション・コード・レジスタ１０４（ＣＣＲ））は、前記選択された条件に基づいて演算結果の性質を表す情報（図２の１０４ｄ（ＤＣ））以外に恒常的に演算結果の性質を表す情報（図２の１０４ａ（Ｎ），１０４ｂ（Ｚ），１０４ｃ（Ｖ）など）を同時に保持する手段を持つことを特徴としている。また、前記条件選択手段（例えば図４の１行目）は、恒常的に演算結果の性質を表す複数の情報の中からひとつを選択することを特徴としている。

【００１２】さらに、前記条件付き演算命令（図５）は、命令コード中に条件を指定するフィールド（５１）を持ち、該条件を指定するフィールド（５１）は、無条件で実行する選択肢（５２の「０」）と、前記選択された条件情報を保持する手段で保持されている条件が真の場合のみ所要の演算処理を実行する選択肢（５２の「１」）を少なくとも持つことを特徴としている。また、前記条件付き演算命令（図１０）は、命令コード中に条件を指定するフィールド（８１）を持ち、該条件を指定するフィールド（８１）は、無条件で実行する選択肢（８２の「００」）と、前記選択された条件情報を保持する手段で保持されている条件が「真」の場合に所要の演算処理を実行する選択肢（８２の「０１」）と、「偽」の場合に所要の演算処理を実行する選択肢（８２の「１０」）とを少なくとも持つことを特徴としている。また、前記選択された条件情報を保持する手段（図１４の１０３ａ（ＣＳ０），１０３ｂ（ＣＳ１），１０３ｃ（ＣＳ２））は、前記演算結果の性質を表す情報を保持する手段（図１４の１０４ａ（Ｎ），１０４ｂ（Ｚ），１０４ｃ（Ｖ）と同一のレジスタ（１５００（ＣＲ））で構成されていることを特徴としている。

【００１３】

【作用】本発明の構成において、条件選択手段（例えば図４の１行目）は選択すべき条件を規定するコード（条件情報）を条件情報保持手段（コンディション・セレクト・レジスタ１０３（ＣＳＲ））に書き込み、以後新たに別の条件情報によって該条件情報が書き換えられるか、またはリセット処理動作などによって初期化されるまで、その条件情報を保持しておく。また、条件付き実行命令用の演算結果の性質を表す情報を保持する保持手段（図２の１０４ｄ）は、コンディション・コードを更新すべき命令が実行される度に条件情報保持手段（コンディション・セレクト・レジスタ１０３（ＣＳＲ））に保持されている情報に基づいてその内容が更新される。一方、各演算命令は、図５に示すように、無条件で実行するか条件付きで実行するかを選択するビットを持ち、条件つきで実行させる選択肢（図５の５２）を持っており、条件付きで演算命令が実行されると、プロセッサは該条件付き実行命令用の演算結果の性質を表す情報を保持する保持手段（コンディション・コード・レジスタ１０４（ＣＣＲ））の内容をチェックしてその演算を実行すべきかどうかを判定する。

【００１４】または、各演算命令は、図１０に示すように、無条件で実行する選択肢（８２の「００」）と、前記選択された条件情報を保持する手段で保持されている条件が「真」の場合に所要の演算処理を実行する選択肢（８２の「０１」）と、「偽」の場合に所要の演算処理を実行する選択肢（８２の「１０」）とを持っており、条件付きで演算命令が実行されると、プロセッサは該条件付き実行命令用の演算結果の性質を表す情報を保持する保持手段（コンディション・コード・レジスタ１０４（ＣＣＲ））の内容をチェックして所要の演算処理を実行するか否かを判定する。この結果、各演算命令に、無条件か条件付きで実行するかを選択するビット、または、各演算命令に、無条件か，条件情報保持手段に保持されている条件が「真」の時に実行するか，「偽」の時に実行するかを選択する少数のビットを付加するだけで済み、各演算命令毎に全ての種類の条件を指定する長いフィールドを設ける必要がなくなり、命令の種類やオペランドの自由度の制限による性能低下を防止し、かつプログラムメモリの小容量化が可能になる。

【００１５】

【実施例】図１は本発明の第１の実施例を説明するための構成図である。同図において、１００は本発明を適用した情報処理装置の演算実行回路、１０１はデータを格納しているメモリ、１０２は演算に必要なソースデータや演算結果を格納するレジスタファイル、１０３は条件付き演算命令に使用するために選択されたコンディション・コード情報を保持するコンディション・セレクト・レジスタ（ＣＳＲ）、１０４は演算結果のコンディション・コードおよび条件付き実行命令用のコンディション・コードを保持するコンディション・コード・レジスタ（ＣＣＲ）、１０５は様々な算術演算や論理演算を実行する算術論理演算器（ＡＬＵ）、１０６は演算実行に必要な制御信号を生成する制御信号生成回路である。

【００１６】また、１０７は制御信号生成回路１０６から送られるメモリ１０１の読み出し，書き込みを制御する制御信号、１０８は制御信号生成回路１０６から送られるコンディション・セレクト・レジスタ１０３（ＣＳＲ）の読み出し，書き込みを制御する制御信号、１０９は制御信号生成回路１０６から送られるレジスタ・ファイル内の各レジスタ毎の、データバス１２１あるいは算術論理演算器１０５とのデータの入出力を制御する信号群、１１０は制御信号生成回路１０６から送られる算術論理演算器１０５で実行する演算を選択して実行させるための制御信号群、１１１はメモリ１０１とデータバス１２１とを接続しているデータバス、１１２はレジスタ

・ファイル１０２とデータバス１２１とを接続しているデータバス、１１３，１１４はレジスタ・ファイル１０２から算術論理演算器１０５へ供給されるソース・データ、１１５は算術論理演算器１０５で実行された演算結果をレジスタ・ファイル１０２へ書き込むべきデスティネーション・データ、１１６は算術論理演算器１０５で実行された演算結果のコンディション・コードをコンディション・コード・レジスタ１０４（ＣＣＲ）に反映させるのに必要なデータ信号群である。

【００１７】さらに、１１７はコンディション・セレクト・レジスタ１０３（ＣＳＲ）とデータバス１２１とを接続しているデータバス、１１８はコンディション・セレクト・レジスタ１０３（ＣＳＲ）に保持された選択されたコンディション・コード情報をコンディション・コード・レジスタ１０４（ＣＣＲ）に伝える選択情報信号、１１９は条件付き演算命令や通常の条件分岐命令の実行制御のために、コンディション・コード・レジスタ１０４（ＣＣＲ）の保持情報を制御信号生成回路１０６に伝えるための信号、１２０はコンディション・コード・レジスタ１０４（ＣＣＲ）とデータバス１２１とを接続しているデータバス、１２１はデータバス、１２２はコンディション・コード・レジスタ１０４（ＣＣＲ）の読み出し，書き込みを制御する信号である。なお、上記の信号は単数または複数の信号を意味し、混乱しない限りにおいて信号と信号線を同一の符号で呼ぶことがある。また本来の情報処理装置に当然含まれる他の要素回路、すなわちメモリ１０１に供給するアドレスおよびアドレス発生回路、命令のデコード回路やフロー制御回路などは本発明とは直接関係がないのでここでは省略している。

【００１８】次に、通常の基本的な演算命令の実行手順を説明する。演算命令が実行されると、まず制御信号生成回路１０６は制御信号１０９を通じて演算に必要なソース・データをレジスタ・ファイル１０２から１１３および１１４に出力させる。情報処理装置が備える命令によっては、演算に必要なソース・データをメモリから供給するように指定する場合もあるが、本発明にはソース・データの出力元がどこであるかは無関係なので、以下の説明ではソース・データをレジスタ・ファイル１０２から供給するものと仮定する。レジスタ・ファイル１０２からソース・データを受け取った算術論理演算器１０５は、制御信号１１０によって指定された演算を実行し、その演算結果をデスティネーション・データ１１５として出力し、信号群１０９で指定されたレジスタ・ファイル位置（デスティネーション・レジスタ）に入力する。この時、同時に演算結果のコンディション・コード情報をデータ信号群１１６を通じてコンディション・コード・レジスタ１０４（ＣＣＲ）に送る。

【００１９】図２にコンディション・セレクト・レジスタ１０３（ＣＳＲ）およびコンディション・コード・レジスタ１０４（ＣＣＲ）、およびその周辺部分の詳細な構成例を示す。図２中、１０３ａ（ＣＳ０），１０３ｂ（ＣＳ１），１０３ｃ（ＣＳ２）はコンディション・セレクト・レジスタ１０３（ＣＳＲ）を構成するフラグビット、２００は上記各フラグビットの情報をデコードして条件付き演算命令で指定する条件をコンディション・コード・レジスタ１０４（ＣＣＲ）に送る選択情報信号１１８を生成する回路（デコーダ）、１０４ａは演算結果が正か負かを示すフラグ（Ｎ），１０４ｂは演算結果がゼロかどうかを示すフラグ（Ｚ），１０４ｃは演算結果がプロセッサが表現できるデータの範囲を超えたことを示すフラグ（Ｖ），１０４ｄは条件付き実行命令用のコンディション・コードを保持するフラグ（ＤＣ）である。

【００２０】また、１１６ａ，１１６ｂ，１１６ｃ，１１６ｄは算術論理演算回路１０５から送られてくる演算結果のコンディション・コードをコンディション・コード・レジスタ１０４（ＣＣＲ）に反映させるのに必要な個別のデータ信号群、２０１はデータ信号群１１６ａ，１１６ｂ，１１６ｃ，１１６ｄおよびコンディション・セレクト・レジスタ１０３（ＣＳＲ）から送られてきた選択情報信号１１８を用いてフラグ１０４ｄに入力すべき状態信号２０２を生成する条件生成／選択回路である。なお、ここでは条件付き実行命令用のコンディション・コードを保持するフラグ１０４ｄ以外の通常のコンディション・コードフラグとして３種類のフラグを例示しているが、この部分の種類およびその数については、特定の数に制限するものではなく、さらに不要とあれば、全くなくてもよいことは明らかである。また、状態信号２０２を生成するための信号１１６の数が、個別に反映するコンディション・コードフラグの数と異なっていてもよいことも明らかである。

【００２１】次に、前述した図１５のフローチャートの処理手順を本発明によるアセンブラ的なニモニックで表現したプログラム例を図４に示す。条件付き演算命令を実行するにあたっては、図４の１行目（SET CSR [2:0] =101）に書かれているように、まずコンディション・セレクト・レジスタ１０３（ＣＳＲ）に選択すべき条件を規定する情報を設定する（条件選択手段）。ここでは、コンディション・セレクト・レジスタ１０３（ＣＳＲ）の第２ビット，第１ビット，第０ビットがそれぞれ１，０，１、すなわち、結果が正数（０を含む）であることを条件としている。前述の条件付き実行命令用のコンディション・コードを保持する手段にどんなコンディションを反映させるかを予め選択する命令として、コンディション・セレクト・レジスタ１０３（ＣＳＲ）内の各ビットにデータを直接セットする特別の命令を新しく設けている。具体的な動作としては、新しいセット命令からデコードされた選択情報をコンディション・セレクト・レジスタ１０３（ＣＳＲ）の各ビット１０３ａ，１０３

ｂ，１０３ｃに対応してコード化し、それをデータバスに出力して制御信号１０８をイネーブルにして書き込む方法が考えられる。

【００２２】しかしコンディション・セレクト・レジスタ１０３（ＣＳＲ）に選択すべき条件を規定する情報を設定する方法は、上述したように特に新しいセット命令を追加しなくても、レジスタ・ファイル１０２またはメモリ１０１に格納されているデータをデータ転送命令によって送り込む方法でも実現できる。またここではコンディション・セレクト・レジスタ１０３（ＣＳＲ）を構成するフラグビットは３ビットで示しているが、ビット数は情報処理装置が条件付き演算命令に必要としている条件の種類を全て区別できればいくらでも構わない。ここでは例として８種類の条件のうちから１つを選択するコンディション・セレクト・レジスタ１０３（ＣＳＲ）を構成する各ビットの定義例を図３に示してある。この定義例では、結果が０でないときは０００，結果が０のときは００１，結果がオーバフローしたときは０１０，結果がオーバフローしていないときは０１１，結果が負数のときは１００，結果が正数（０を含む）のときは１０１，キャリーが生成されたときは１１０，キャリーが生成されなかったときは１１１としている。

【００２３】コンディション・セレクト・レジスタ１０３（ＣＳＲ）の各ビット１０３ａ（ＣＳ０），１０３ｂ（ＣＳ１），１０３ｃ（ＣＳ２）に書き込まれた情報は、以後新たに別の情報によって書き換えられるか、またはリセット処理動作などによって初期化されるまで保持され続ける。コンディション・セレクト・レジスタ１０３（ＣＳＲ）の各ビット１０３ａ，１０３ｂ，１０３ｃに選択すべき条件を規定する情報が設定されると、その情報をデコーダ２００で選択されたコンディション・コード情報をコンディション・コード・レジスタ１０４（ＣＣＲ）に伝える選択情報信号１１８に変換する。この変換過程は必ずしも必要という訳ではなく、１０３ａ，１０３ｂ，１０３ｃに保持されている状態をそのまま選択情報信号１１８として出力しても構わない。しかしその場合は、コンディション・コード・レジスタ１０４（ＣＣＲ）内のビット１０４ｄに格納する状態信号２０２を生成する全ての処理を条件生成／選択回路２０１で行うことになる。

【００２４】コンディション・セレクト・レジスタ１０３（ＣＳＲ）の各ビット１０３ａ，１０３ｂ，１０３ｃに選択すべき条件を規定する情報が設定された後、コンディション・コード・レジスタ１０４（ＣＣＲ）内のビット１０４ｄは、コンディション・コードが更新される可能性のある演算命令が実行される度にコンディション・セレクト・レジスタ１０３（ＣＳＲ）に保持されている情報に基づいてその内容が更新される。すなわち、図４の２行目に書かれているように、コンディション・コードが更新される可能性のある演算命令（inst.1）が算術論理演算器１０５で実行されると、コンディション・コード・レジスタ１０４（ＣＣＲ）内の条件生成／選択回路２０１は、算術論理演算回路１０５から送られてきた様々な演算結果のコンディション・コード情報１１６と、コンディション・セレクト・レジスタ１０３（ＣＳＲ）からの選択情報信号１１８によってフラグ１０４ｄに入力すべき状態信号２０２を生成する。

【００２５】図１５の処理手順では次に演算結果が正数（０を含む）であるかどうかを条件とした演算命令が必要であるので、図３に示された正数条件コード「１０１」が各ビット１０３ａ，１０３ｂ，１０３ｃに保持されており、条件生成／選択回路２０１は演算結果のコンディション・コード情報１１６のうちの信号１１６ａの反転情報を状態信号２０２として出力する。すなわちネガティブ信号１１６ａが「偽」のとき、状態信号２０２は「真」状態を出力する。状態信号２０２はそのままフラグ１０４ｄに入力される。コンディション・コード・レジスタ１０４（ＣＣＲ）内の他のフラグ１０４ａ，１０４ｂ，１０４ｃにはそれぞれ信号１１６ａ，１１６ｂ，１１６ｃが入力される。続いて、図４の３行目の条件付き演算命令（if P inst.2）が実行されると、前述したように２行目の演算命令の演算結果が正数（０を含む）かどうかが既にフラグ１０４ｄに保持されているので、フラグ１０４ｄが「真」状態ならば指定された演算を実行し、「偽」状態ならば実行しないでそのまま次の４行目の命令（ｉｎｓｔ．３）の実行に移る。

【００２６】図５に本発明における条件付き演算命令の一例を示す。同図において、５０は命令コード、５１は条件指定フィールド、５２は条件指定フィールドの具体的なコード定義例を示している。ここでは、命令コード中に条件を指定する条件指定フィールドとして１ビット設け、該１ビットの条件指定フィールドの値によって該命令を無条件に実行するのか（０の場合）、条件付きで実行するのか（１の場合）を指定した例である。条件付きで実行するとは、選択された条件情報を保持する手段（条件情報保持手段：コンディション・セレクト・レジスタ１０３（ＣＳＲ））に保持されている条件が真の場合にのみ所要の演算処理を実行することを意味している。

【００２７】図１５では、与えられた条件が「真」の時のみ追加的な処理を実行し、「偽」の時は追加処理を行わないフローを示しているが、図６に示すように「偽」の時も別の処理がある場合が考えられる。図６の処理手順をアセンブラ的なニモニックで表現したプログラム例を図７に示す。図６において、まずステップ６０で算術演算１を実行し（図７の１行目）、ステップ６１でその演算結果を判定し、演算結果が正数（０を含む）であれば（すなわち負数でなければ）ステップ６２で動作２ａを実行し（図７の３行目）、演算結果が正数（０を含む）でなければ（すなわち負数であれば）ステップ６３

でステップ６２とは異なる動作２ｂを実行し（図７の５行目）、ステップ６２またはステップ６３を実行した後にステップ６４において動作３を実行する（図７の６行目）。図７のプログラム例では与えられた条件の真／偽に関わらず、必ず１回は分岐命令を実行することになる。

【００２８】このような例についても、本発明を適用することが可能である。本発明による図６の処理手順をアセンブラ的なニモニックで表現したプログラム例を図８に示す。図８のプログラム例によると、その２行目（if P inst.2a）および３行目（if N inst.2b）に示されているように分岐命令を用いず条件付き演算命令を用いている。ここで注意すべきことは、図８の２行目の条件付き演算命令（if P inst.2a）が条件が「真」で実行された時、演算結果によってレジスタ１０４（ＣＣＲ）のコンディション・コードを更新しないようにすることである。この制御は非常に容易であり、例えば条件付き演算命令が実行された時は制御信号１２２をディセーブル状態にしておくことにより、コンディション・コードの更新が行なわれないようにすることができる。このような制御は、前述した図１５のような処理において同様に行なっても全く問題ないので、無条件演算実行時はコンディション・コードを更新することとし、条件付き演算実行時は条件の成立／不成立に関わらずコンディション・コードを更新しないことにすれば、図１５および図６の双方の処理を本発明を用いて同一のプロセッサで実現できる。

【００２９】図９は、条件付き演算命令の他の例であり、複数の条件を設定するとともに、命令中に条件を指定する条件指定フィールド７１を設けたものである。この条件指定フィールドの値で指定する所定の条件が「真」のときに対応する所要の演算を行うようにしたものである。条件指定フィールド７１のコード定義例としては、例えば、図９の７２に示すように、無条件で実行する選択肢（７２の「００」）と、前記選択された条件情報を保持する手段で保持されている条件１が「真」の場合に所要の演算処理を実行する選択肢（７２の「０１」）と、前記選択された条件情報を保持する手段で保持されている条件２が「真」の場合に所要の演算処理を実行する選択肢（７２の「１０」）と、前記選択された条件情報を保持する手段で保持されている条件３が「真」の場合に所要の演算処理を実行する選択肢（７２の「１１」）とを有している。

【００３０】また、図１０は、条件付き演算命令の別の例である。この条件付き演算命令は命令中に条件を指定する条件指定フィールド８１を持っている。条件指定フィールド８１のコード定義例としては、例えば、同図８２に示すように、無条件で実行する選択肢（８２の「００」）と、前記選択された条件情報を保持する手段で保持されている条件１が「真」の場合に所要の演算処理を実行する選択肢（８２の「０１」）と、保持されている条件１が「偽」の場合に所要の演算処理を実行する選択肢（８２の「１０」）とを有している。

【００３１】本発明の実施例は他にも様々な形式が考えられる。例えば図２の別の実施例として図１１に示すように、図２の条件生成／選択回路２０１をセレクタ１１００で構成し、算術論理演算回路１０５から送られてくる信号群１１６の中の１つを該セレクタ１１００で選択して信号２０２として出力し、フラグ１０４ｄに入力させてもよい。この場合、コンディション・セレクト・レジスタ１０３（ＣＳＲ）のコード定義例としては、図１２に示すように信号群１１６のどれを選ぶかを指定するようにすればよい。図１２の定義例では、ゼロ判定信号選択のときは００，オーバフロー判定信号選択のときは０１，正負判定信号選択のときは１０，キャリー発生判定信号選択のときは１１である。図１１の実施例では、コンディション・セレクト・レジスタ１０３（ＣＳＲ）は２ビットのレジスタになっているが、ビット数についてはこの実施例では特に２ビットでなければならないという訳ではなく、例えばより多くのコンディションから選びたい場合にはもっとビット数を増やして構わないことは明白である。

【００３２】コンディション・セレクト・レジスタ１０３（ＣＳＲ）やコンディション・コード・レジスタ１０４（ＣＣＲ）の構成方法についても様々な方法が考えられる。図１３および図１４は、両者を単一のレジスタで構成した一実施例である。図１３中、１５００は図１のコンディション・セレクト・レジスタ１０３（ＣＳＲ）とコンディション・コード・レジスタ１０４（ＣＣＲ）を合体させたコントロール・レジスタＣＲであり、１５０１はコントロール・レジスタ１５００とデータバス１２１とを接続しているデータバス、１５０２はコントロール・レジスタ１５００と制御信号生成回路１０６とを接続しているデータバスである。残りは図１および図２と同一である。この実施例は、コンディション・セレクト用のビットとコンディション・コードのビットが同一のレジスタ内でビット位置が異なる以外、具体的な動作は先述の第一の実施例と同様である。

【００３３】以上の実施例では、コンディション・セレクト・レジスタ１０３（ＣＳＲ）、コンディション・コード・レジスタ１０４（ＣＣＲ）、コントロール・レジスタ１５００（ＣＲ）はすべて通常のレジスタであることを前提として述べてきたが、例えばメモリマップ上にマッピングされたレジスタであってもよいことは明らかである。

【００３４】

【発明の効果】以上述べたように、本発明によれば、通常のコンディション・コード・フラグ以外に条件付き実行命令用のコンディション・コードを保持する手段を設け、該条件付き実行命令用のコンディション・コードを

保持する手段にどんなコンディションを反映させるかを予め選択する命令およびその選択情報を保持する手段を設けることにより、命令の種類やオペランドの自由度の制限による性能低下、または命令コード長の増大によるプログラムメモリ容量増加などの問題発生することなく処理効率の向上を図ることができる条件付き演算命令をサポートすることができる。

【図面の簡単な説明】

【図１】本発明の第１の実施例である。

【図２】本発明の第１の実施例の部分的な詳細ブロック図である。

【図３】コンディション・セレクト・レジスタの各ビットの定義例である。

【図４】本発明を用いて図１５の処理内容をアセンブラ的なニモニックで表現したプログラム例である。

【図５】１ビットの条件指定フィールドを持つ命令コード例である。

【図６】条件付き実行動作を含む第２のフローチャート図である。

【図７】図６の処理内容をアセンブラ的なニモニックで表現したプログラム例である。

【図８】本発明を用いて図６の処理内容をアセンブラ的なニモニックで表現したプログラム例である。

【図９】２ビットの条件指定フィールドを持つ命令コード例である。

【図１０】２ビットの条件指定フィールドを持つ第２の命令コード例である。

【図１１】本発明の図１の実施例の別な部分的な詳細ブロック図である。

【図１２】コンディション・セレクト・レジスタの各ビットの第２の定義例である。

【図１３】本発明の第２の実施例を示す図である。

【図１４】本発明の第２の実施例の部分的な詳細ブロック図である。

【図１５】条件付き実行動作を含むフローチャート例である。

【図１６】図１５の処理内容をアセンブラ的なニモニックで表現したプログラム例である。

【図１７】図１５の処理内容をアセンブラ的なニモニックで表現した別のプログラム例である。

【図１８】３ビットの条件指定フィールドを持つ命令コード例である。

【符号の説明】

１００：本発明を適用した情報処理装置の演算実行回路

１０１：データを格納しているメモリ

１０２：演算に必要なソースデータや演算結果を格納するレジスタファイル

１０３：条件付き演算命令に使用するために選択されたコンディション・コード情報を保持するコンディション・セレクト・レジスタ（CSR）

１０４：演算結果のコンディション・コードおよび条件付き実行命令用のコンディション・コードを保持するコンディション・コード・レジスタ（CCR）

１０５：様々な算術演算や論理演算を実行する演算器（ALU）

１０６：演算実行に必要な制御信号を生成する制御信号生成回路

１０７：メモリ１０１の読み出し，書き込みを制御する信号

１０８：コンディション・セレクト・レジスタ１０３（ＣＳＲ）の読み出し，書き込みを制御する信号

１０９：レジスタ・ファイル内の各レジスタ毎のデータバス１２１あるいは算術論理演算器１０５とのデータの入出力を制御する信号群

１１０：算術論理演算器１０５で実行する演算を選択し，実行させるための制御信号群

１１１：メモリ１０１とデータバス１２１とを接続しているデータパス

１１２：レジスタ・ファイル１０２とデータバス１２１とを接続しているデータパス

１１３，１１４：レジスタ・ファイル１０２から算術論理演算器１０５へ供給されるソース・データ

１１５：算術論理演算器１０５で実行された演算結果をレジスタ・ファイル１０２へ書き込むべきデスティネーション・データ

１１６：算術論理演算器１０５で実行された演算結果のコンディション・コードをコンディション・コード・レジスタ１０４（ＣＣＲ）に反映させるのに必要なデータ信号群

１１７：コンディション・セレクト・レジスタ１０３（ＣＳＲ）とデータバス１２１とを接続しているデータパス

１１８：コンディション・セレクト・レジスタ１０３（ＣＳＲ）に保持された選択されたコンディション・コード情報をコンディション・コード・レジスタ１０４（ＣＣＲ）に伝える信号

１１９：条件付き演算命令や通常の条件分岐命令の実行制御のために、コンディション・コード・レジスタ１０４（ＣＣＲ）の保持情報を制御信号生成回路１０６に伝えるための信号

１２０：コンディション・コード・レジスタ１０４（ＣＣＲ）とデータバス１２１とを接続しているデータパス

１２１：データバス

１２２：コンディション・コード・レジスタ１０４（ＣＣＲ）の読み出し，書き込みを制御する信号

１０３ａ，１０３ｂ，１０３ｃ：コンディション・セレクト・レジスタ１０３を構成するフラグビット

１０４ａ：演算結果が正か負かを示すフラグ（N）

１０４ｂ：演算結果がゼロかどうかを示すフラグ（Z）

１０４ｃ：演算結果がプロセッサが表現できるデータの

15

範囲を超えたことを示すフラグ（V）

１０４ｄ：条件付き実行命令用のコンディション・コードを保持するフラグ

１１６ａ，１１６ｂ，１１６ｃ，１１６ｄ：算術論理演算回路１０５から送られてくる演算結果のコンディション・コードをコンディション・コード・レジスタ１０４（ＣＣＲ）に反映させるのに必要な個別のデータ信号

２００：各フラグビットの情報をデコードして条件付き演算命令で指定する条件をコンディション・コード・レジスタ１０４（ＣＣＲ）に送る選択情報信号１１８を生成する回路

２０１：データ信号群１１６ａ，１１６ｂ，１１６ｃ，

16

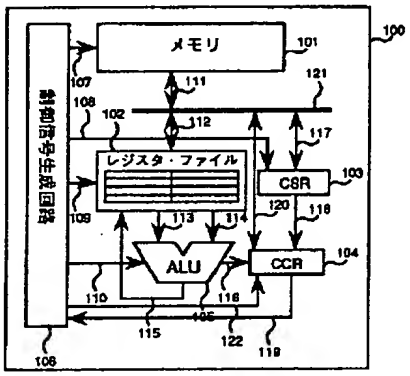
１１６ｄおよびコンディション・セレクト・レジスタ１０３（ＣＳＲ）から送られてきた選択情報信号１１８を用いてフラグ１０４ｄに入力すべき状態信号２０２を生成する条件生成／選択回路

１５００：コンディション・セレクト・ビット，演算結果のコンディション・コードおよび条件付き実行命令用のコンディション・コードを保持するコントロール・レジスタ（CR）

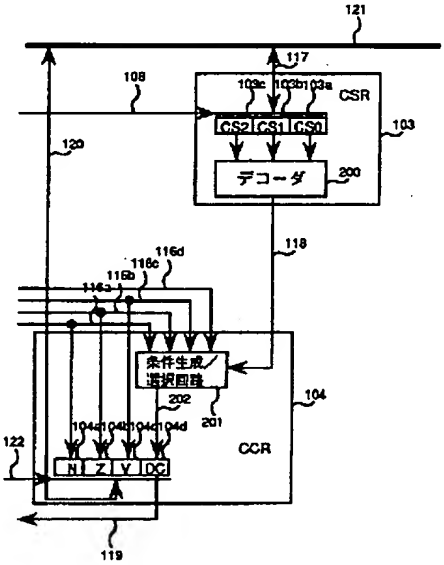
１５０１：コントロール・レジスタ１５００とデータバス１２１とを接続しているデータパス

１５０２：コントロール・レジスタ１５００の読み出し，書き込みを制御する信号

【図１】



【図２】



【図４】

```
1     SBT CSR[2:0]=101
2     inst. 1
3     if P inst. 2
4     inst. 3
5
6
```

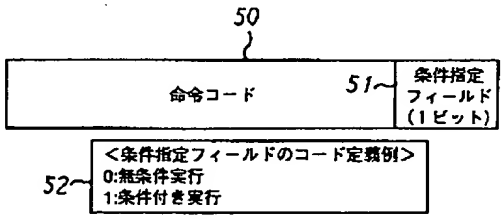
【図１６】

```
1     inst. 1
2     if N goto L1
3     inst. 2
4  L1:inst. 3
5
6
```

【図３】

<コンディション，セレクト・レジスタ１０３のコード定義例>
000:結果がゼロでない時
001:結果がゼロの時
010:結果がオーバフローした時
011:結果がオーバフローしていない時
100:結果が負数の時
101:結果が正数の時
110:キャリーが生成された時
111:キャリーが生成されなかった時

【図５】



【図７】

```
1     inst. 1
2     if N goto L1
3     inst. 2a
4     goto L2
5  L1:inst. 2b
6  L2:inst. 3
7        |
8        |
```

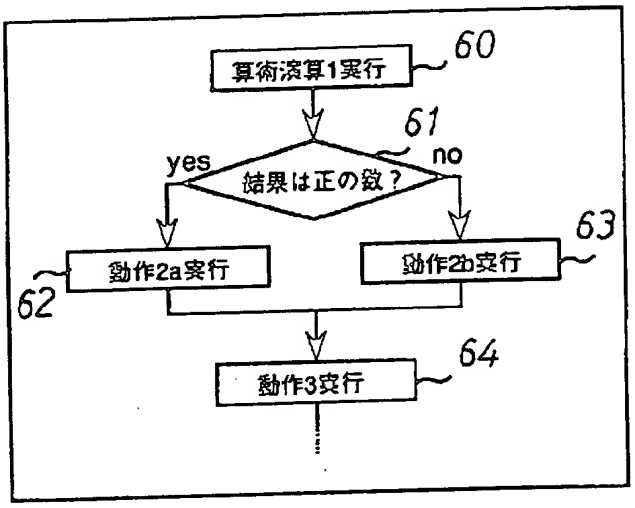
【図８】

```
1     inst. 1
2     if P inst. 2a
3     if N inst. 2b
4     inst. 3
5        |
6        |
```

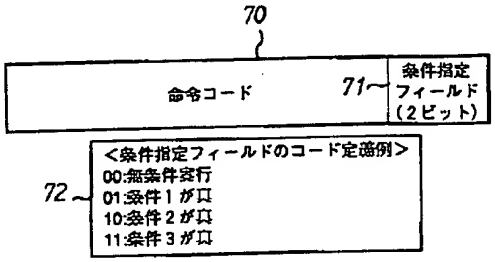
【図１２】

<コンディション，セレクト・レジスタ１０３のコード定義例>
00:ゼロ判定信号選択
01:オーバフロー判定信号選択
10:正負判定信号選択
11:キャリー発生判定信号選択

【図６】



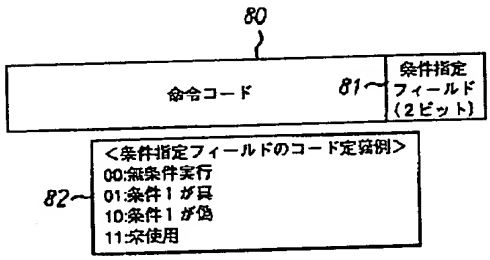
【図９】



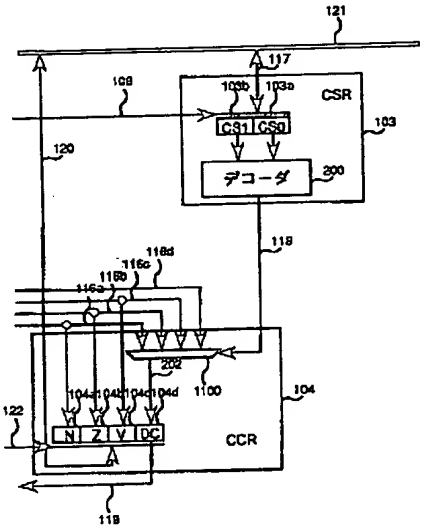
【図１７】

```
1    inst. 1
2    if P inst. 2
3    inst. 3
4
5
6
```

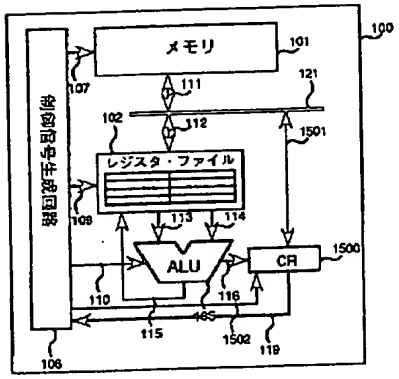
【図１０】



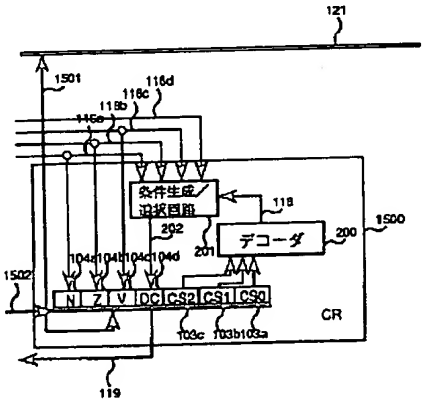
【図１１】



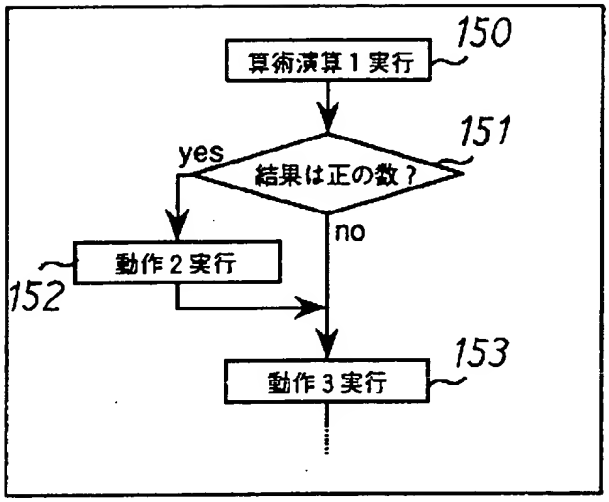
【図１３】



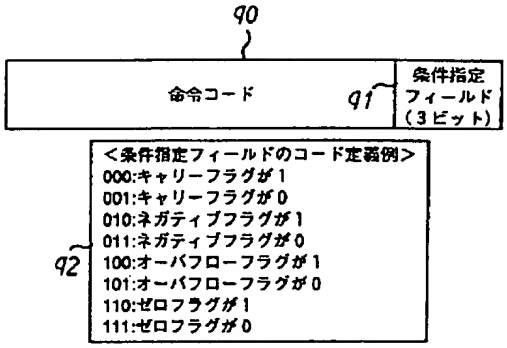
【図１４】



【図１５】


算術演算１実行
150
結果は正の数？
151
yes
no
動作２実行
152
動作３実行
153

【図１８】


90
命令コード
91
条件指定
フィールド
（３ビット）
<条件指定フィールドのコード定義例>
000:キャリーフラグが１
001:キャリーフラグが０
010:ネガティブフラグが１
011:ネガティブフラグが０
100:オーバフローフラグが１
101:オーバフローフラグが０
110:ゼロフラグが１
111:ゼロフラグが０
92